19. Enkele toepassingen

Dobbelstenen

Een dobbelsteen geeft een willekeurig getal van 1 tot 6 weer. Dat kan met puntjes op een kubus of elektronisch. Wij gebruiken de laatste vorm. We hebben een controller nodig, een duidelijk leesbaar display en een stukje software. In het voorbeeld hieronder gebruiken we het serieel 7-segment display.

```
#------------------------------#
#  dobbelsteen.py              #
#                              #
#  23 maart 2021               #
#  Dirk Ghysels                #
#------------------------------#
import tm1637
from machine import Pin
from utime import sleep
import random
mydisplay = tm1637.TM1637(clk=Pin(1), dio=Pin(0))
mydisplay.show("    ")
for k in range(10):
    x = random.randint(1,6)
    y = random.randint(1,6)
    strxy = str(x)+"  "+str(y)  # 2 dobbelstenen
#     strxy = str(x)  # 1 dobbelsteen
    mydisplay.show(strxy)
    sleep(.2)
```

Dit programma toont één of twee dobbelstenen, dat pas je aan in de regel waar strxy gedefinieerd is. Eerst zie je de dobbelstenen uitrollen tot er twee cijfers blijven. Kopieer het programma naar de controller met naam main.py. Het start automatisch op. Je start een nieuwe sessie door het programma te resetten, de Pimoroni heeft een reset-toets, voor de Pico maak je een reset-toets met een drukknop tussen GND en aansluiting **RUN**.

Afstand meten met HC-SR04

Afstanden meten we met module HC-SR04. Deze heeft twee ultrasone transducers: de ene stuurt een puls ultrasoon geluid (een onhoorbaar hoge frekwentie), de andere ontvangt het. Met het tijdsverschil tussen zenden en onvangen en me de geluidssnelheid kunnen we de afstand berekenen De geluidssnelheid hangt af van de temperatuur, een bruikbare gemiddelde waarde is 343 m/sec.



Figuur 115: module voor afstandsmeting

Het programma hieronder is gebaseerd op een artikel uitTom's Hardware.
https://www.tomshardware.com/how-to/raspberry-pi-pico-ultrasonic-sensor

```
#-----------------------------#
#  afstand-v3.py              #
#                             #
#  25 maart 2021              #
#  Dirk Ghysels               #
#-----------------------------#
from machine import Pin
import utime

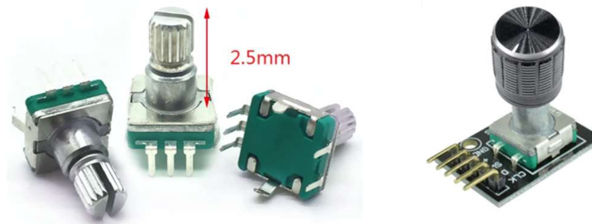
# Init HC-SR04 pinsfrom machine import Pin
import utime
trigger = Pin(3, Pin.OUT)
echo = Pin(2, Pin.IN)

def ultra():
    trigger.low()
    utime.sleep_us(2)
    trigger.high()
    utime.sleep_us(5)
    trigger.low()
    while echo.value() == 0:
        signaloff = utime.ticks_us()
    while echo.value() == 1:
        signalon = utime.ticks_us()
    timepassed = utime.ticks_diff(signalon, signaloff)
    distance = (timepassed * 0.0343) / 2
    return distance

while True:
    afstand = ultra()
    print (afstand)
```

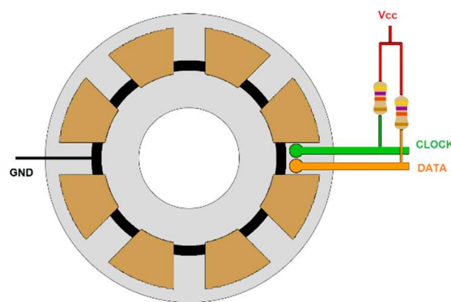
Rotary encoders

Een **rotary decoder** of een **draai-encoder**, is een positie-sensor die een signaal afgeeft, afhankelijk van een draaibeweging. Een relatieve rotary decoder geeft aleen de richting van de draaiing weer (wijzerin of tegenwijzerzin) en de hoeveel draaiing. De juiste eindstand ken je niet. Een toepassing is de volumeknop van een versterker. Draai naar links en het geluid wordt zachter. Draai naar rechts en het wordt luider.



Figuur 116: rotary encoders

De foto hierboven toont een eenvoudig model, een mechanische draai-encoder. Hij lijkt op een potentiometer maar dat is hij zeker niet. Hij detecteert draaistappen en draairichting. Er is een schakelaar ingebouwd. Die laatste activeer je door de draaiknop in te drukken. Ze zijn ook verkrijgbaar als een module op een printje, inclusief de pull-up weerstanden. Het binnenwerk is hieronder geschetst:



Figuur 117: mechanische rotary encoder

De acht kopervlakken zijn verbonden met GND en draaien mee met de as van het systeem. Twee sleepcontacten, CLOCK en DATA zijn via pull-up weerstanden verbonden met Vcc. Het niveau op die lijnen is O als ze contact hebben met een kopervlak en 1 zonder contact. Bij draaiing in tegenwijzerzin is clock eerst 0, dan data. Het omgekeerde zien we bij een draaiing in wijzerzin. De draaihoek bepalen we door het aantal overgangen te tellen.

We gebruiken de decoder om een keuze te maken uit een lijst. In het voorbeeld zijn dat getallen tussen min en max, het keuzegetal is teller. In één richting draaien geeft een groter teller en omgekeerd. In het programma vragen we in een oneindige lus de waarde van clock op. Als die verandert kijken we naar de waarde van data.

- Als data ≠ clock, dan is clock eerst gewijzigd . We verlagen teller
- Als data = clock, dan is data eerst gewijzigd . We verhogen teller

Wat er gebeurt na bij het verhogen van de hoogste tellerwaarde hangt af van variabele *periodiek*. Een periodieke teller begint na de hoogste teller terug met de

laagste. Een niet periodiek systeem gaat blijft staan bij de hoogste waarde, we kunnen alleen terugdraaien.

We verbinden de decoder met de RP2040: GND met GND, clock met GPIO2 en data met GPIO4. Het knooppunt van de weerstanden met 3,3 volt van de RP2040.

```
#------------------------------#
#  rotary-encoder.py           #
#                              #
#  28 maart 2021               #
#  Dirk Ghysels                #
#------------------------------#
from machine import Pin
import utime
clock = machine.Pin(2, Pin.IN, Pin.PULL_UP)
data = machine.Pin(4, Pin.IN, Pin.PULL_UP)
mint = 1
maxt = 9
teller = mint
periodiek = False
statusOud = clock.value()
while True:
    status = clock.value()
    if (status != statusOud):
        if (data.value() != status):
            if teller > mint:
                teller -= 1
            else:
                if periodiek == True:
                    teller = maxt
        else:
            if teller < maxt:
                teller += 1
            else:
                if periodiek == True:
                    teller = mint

        print ("teller = %d" %teller)
        statusOud = status
```

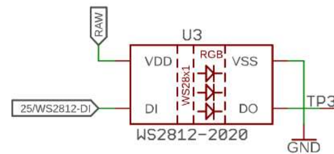
```
Shell
teller = 6
teller = 7
teller = 8
teller = 9
teller = 8
teller = 9
```

Figuur 118: de draaidecoder in actie

Neopixel LED’s

Een neopixel LED of een WS2812 is een slimme RGB-LED met ingebouwde controller chip. Ze kunnen aan elkaar gekoppeld worden met één datalijn. Zo kan je met één

GPIO-aansluiting een hele reeks LED's aansturen. Een neopixel heeft 4 aansluitingen: Vdd (voeding), Vss (massa), DI (ingang) en D0 (uitgang). De ingang sluit je aan op een GPIO-poort, de uitgang verbind je met de ingang van de volgende neopixel.



Figuur 119: de neopixel op de Sparkfun Pro micro: schema

Neopixels zijn apart te koop of in een module met meerdere LED's.



Figuur 120: neopixels en neopixel modules

De aansturing van Neopixel-modules is een standaard library van MicroPython. Het voorbeeld hieronder stuurt een één neopixel aan op poort 21 van een esp32. Het programma kan je aanpassen voor een module met meer LED's.

```
#------------------------------#
#  neopixel.py                 #
#  esp32 - 1 Led               #
#                              #
#  13 december 2021            #
#  Dirk Ghysels                #
#------------------------------#
from machine import Pin
from neopixel import NeoPixel
from utime import sleep
n = 0.5
pin = Pin(21, Pin.OUT)   #
np = NeoPixel(pin, 1)   # NeoPixel driver op GPIO21 met 1 pixels
np[0] = (255, 0,0) # eerste pixel rood
np.write()              # write data to all pixels
#r, g, b = np[0]         # geef de pixelkleur
while (True):
    np[0] = (255, 0, 0)  # rood
    np.write()
    sleep(n)
    np[0] = (0, 255, 0)  # groen
    np.write()
```

```
    sleep(n)
    np[0] = (0, 0, 255)  # blauw
    np.write()
    sleep(n)
    np[0] = (0, 255, 255)  # magenta
    np.write()
    sleep(n)
    np[0] = (255, 0, 255)  # cyaan
    np.write()
    sleep(n)
    np[0] = (255, 255, 0)  # geel
    np.write()
    sleep(n)
    np[0] = (255, 255, 255)  # wit
    np.write()
    sleep(n)
```

Helaas, deze module maakt nog geen deel uit van MicroPython voor de RP2040 of de SAMD21. Een alternatief vind je op de Github pagina van Shreyas Kulkarni:

https://github.com/shreyask21/neopixel_rp2040

Kopieer bestand neopixel_rp2040.py naar de controller.

Hieronder zie je het voorbeeldprogramma uit de library (example.py). Regel 12 moet je aanpassen:

led = neopixel_rp2040.neopixel(LEDS=1, PIN=25)

LEDS is het aantal leds in het neopixel-systeem, PIN is de gebruikte poort voor de neopixels. Het programma is aangepast voor de ingebouwde neopixel van een Sparkfun pro micro rp2040.

```
import utime

''' Import Driver Library '''
import neopixel_rp2040


'''
    Create Object
    Here 'LEDS=' sets the number of LEDs connected in Neopixel string
         'PIN=' is the Pin number used to connect the DIN pin of the
Neopixel to the Raspberry Pi Pico
'''
led = neopixel_rp2040.neopixel(LEDS=1, PIN=25)

'''
    Test All Connected LEDs
'''
led.test()
utime.sleep(2)

'''
    Set Single LED to green color with 50% brightness
```

```
'''
led.set(LED_NUMBER=0, COLOR=led.GREEN, BRIGHTNESS=0.5)
utime.sleep(2)

'''
    Reset Single LED
'''
led.reset(LED_NUMBER=0)
utime.sleep(2)

'''
    Turn On All LEDS to White
'''
led.set()
utime.sleep(2)

'''
    Reset All LEDs
'''
led.reset()
utime.sleep(2)

'''
   Change Brightness of single LED
'''
led.set(LED_NUMBER=0)
led.setBrightness(LED_NUMBER=0, BRIGHTNESS=0.5)
utime.sleep(2)
```

Een neopixel module heeft 3 aansluitingen: GND, voeding en signaal. De signaal-aansluiting verbind je met een GPIO-pin. Het nummer van die pin geeft je aan variabele PIN. GND en voeding verbind je met GND en 5 Volt van de controller.

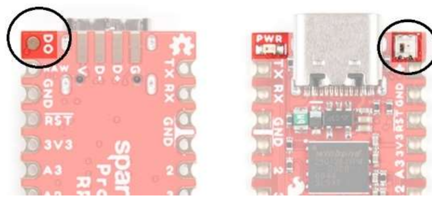
De Adafruit Qt Py RP2040 heeft een neopixel. De voeding van deze neopixel is verbonden met GPIO11, de signaalingang met GPIO12. Maak van GPIO11 een uitgang met waarde 1 om de neopixel te voeden. Hieronder zie je de aanpassingen in het programma

```
import utime
import neopixel_rp2040

from machine import Pin
npin = Pin(11, Pin.OUT)
npin.value(1)

# PIN=' is the Pin number used to connect the DIN pin of the Neopixel to
the Raspberry Pi Pico
led = neopixel_rp2040.neopixel(LEDS=1, PIN=12)
```

De Sparkfun Pro Micro RP2040 heeft één neopixel, verbonden aan GPIO25. Uitgang DO is bereikbaar op een soldeereilandje aan de achterzijde van de print. Hier kan je andere Neopixels aansluiten.



Figuur 121: de neopixel op de Sparkfun Pro micro